AN10866

LPC1700 secondary USB bootloader

Rev. 01 — 8 September 2009 Application note

Document information

Info	Content
Keywords	LPC1700, Secondary USB Bootloader, ISP, IAP
Abstract	This application note describes how to add a custom secondary USB bootloader to a LPC1700 series microcontroller.

NXP founded by Philips

Revision history

Rev	Date	Description
01	20090908	Initial version.

Contact information

For additional information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

© NXP B.V. 2009. All rights reserved.

1. Introduction

NXP's LPC1700 microcontrollers provide the user a convenient way to update the flash contents in the field for bug fixes or product updates. This can be achieved using the following two methods:

- In-System Programming: In-System programming (ISP) is programming or reprogramming the on-chip flash memory, using the boot loader software and UART0 serial port. This can be done when the part resides in the end-user board.
- In Application Programming: In-Application (IAP) programming is performing erase and write operation on the on-chip flash memory, as directed by the end-user application code.

A secondary bootloader is a piece of code which allows a user application code to be downloaded using alternative channels other than the standard UART0 used by the primary bootloader (on-chip). The primary bootloader is the firmware that resides in a microcontroller's boot ROM block and is executed on power-up and resets. After the boot ROM's execution the secondary bootloader is executed. The secondary bootloader in turn will then execute the end-user application.

This application note uses USB as an example for developing a secondary bootloader on a LPC1700 series microcontroller.

2. Requirements

2.1 Hardware requirements

The secondary USB Bootloader application has been developed and tested using a:

- Keil's MCB1700 development board
- MS Windows-based workstation with an available USB port.

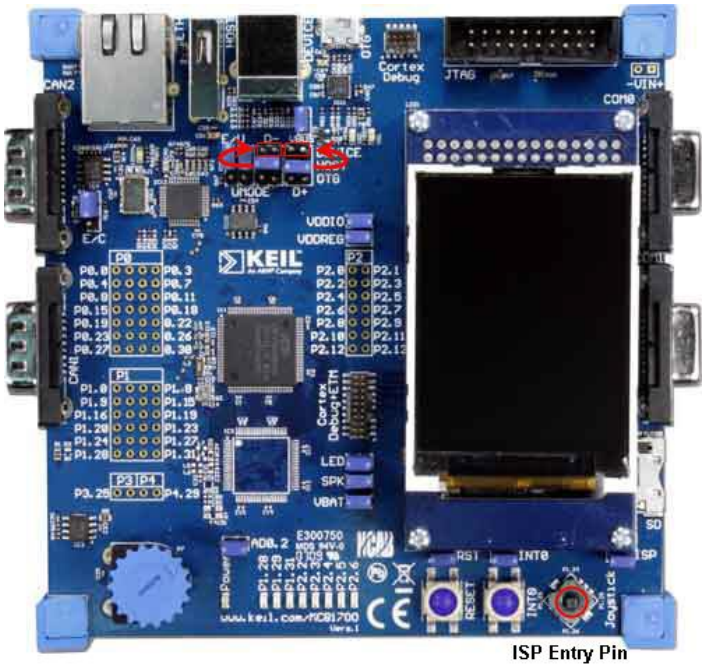


Fig 1. MCB1700 featuring a LPC176x series microcontroller

© NXP B.V. 2009. All rights reserved.

2.1.1 Programming

Programming the bootloader can be done using Keil's U-LINK JTAG module. However, as an additional option the LPC1700's ISP programming functionality can be used instead. To program using the ISP protocol an RS-232 serial cable is required and a free ISP programming tool is available from FlashMagic at: http://www.nxp.com/redirect/flashmagictool.com/

2.1.2 Jumper settings

In order to use the USB device functionality, ensure that the USB's D+ and D- jumpers are both set to the "DEVICE" header pins as shown in Fig 1.

2.1.3 USB cable

A USB-A to USB-B cable is required to power and connect the windows workstation to the MCB1700 development board.

2.1.4 Bootloader's ISP entry mechanism

Having an entry pin grounded during reset will cause the secondary USB bootloader to go into ISP mode. For convenience, P1.20 is used by default because it is wired to the center button of the joy-stick as shown in Fig 1. By pressing down on the MCB1700's joystick during reset will cause the secondary USB bootloader to enter ISP mode.

2.2 Software requirements

An evaluation version of Keil's uVision3 is sufficient to compile the secondary USB bootloader and the sample user applications.

The secondary USB bootloader has been developed and tested using a MS windows-based workstation.

3. Flashing the LPC1700

3.1 Flash sectors

Depending on the LPC1700 part number, the user has up to 512kB (Bytes) of on-chip flash available. This user flash space is divided up into sectors.

In order to make any modifications (even if it is just one byte) to a particular sector, the entire sector must be first erased and then (re)written.

© NXP B.V. 2009. All rights reserved.

Sector Number	Sector Size [kB]	Start Address	End Address	32 kB Part	64 kB Part	128 kB Part	256 kB Part	512 kB Part
0	4	0X0000 0000	0X0000 0FFF	x	x	x	x	x
1	4	0X0000 1000	0X0000 1FFF	x	x	x	x	x
2	4	0X0000 2000	0X0000 2FFF	x	x	x	x	x
3	4	0X0000 3000	0X0000 3FFF	x	x	x	x	x
4	4	0X0000 4000	0X0000 4FFF	x	x	x	x	x
5	4	0X0000 5000	0X0000 5FFF	x	x	x	x	x
6	4	0X0000 6000	0X0000 6FFF	x	x	x	x	x
7	4	0X0000 7000	0X0000 7FFF	x	x	x	x	x
8	4	0x0000 8000	0X0000 8FFF		x	x	x	x
9	4	0x0000 9000	0X0000 9FFF		x	x	x	x
10 (0x0A)	4	0x0000 A000	0X0000 AFFF		x	x	x	x
11 (0x0B)	4	0x0000 B000	0X0000 BFFF		x	x	x	x
12 (0x0C)	4	0x0000 C000	0X0000 CFFF		x	x	x	x
13 (0x0D)	4	0x0000 D000	0X0000 DFFF		x	x	x	x
14 (0X0E)	4	0x0000 E000	0X0000 EFFF		x	x	x	x
15 (0x0F)	4	0x0000 F000	0X0000 FFFF		x	x	x	x
16 (0x10)	32	0x0001 0000	0x0001 7FFF			x	x	x
17 (0x11)	32	0x0001 8000	0x0001 FFFF			x	x	x
18 (0x12)	32	0x0002 0000	0x0002 7FFF				x	x
19 (0x13)	32	0x0002 8000	0x0002 FFFF				x	x
20 (0x14)	32	0x0003 0000	0x0003 7FFF				x	x
21 (0x15)	32	0x0003 8000	0x0003 FFFF				x	x
22 (0x16)	32	0x0004 0000	0x0004 7FFF					x
23 (0x17)	32	0x0004 8000	0x0004 FFFF					x
24 (0x18)	32	0x0005 0000	0x0005 7FFF					x
25 (0x19)	32	0x0005 8000	0x0005 FFFF					x
26 (0x1A)	32	0x0006 0000	0x0006 7FFF					x
27 (0x1B)	32	0x0006 8000	0x0006 FFFF					x
28 (0x1C)	32	0x0007 0000	0x0007 7FFF					x
29 (0x1D)	32	0x0007 8000	0x0007 FFFF					x

(1) Flash space varies in size, depending on the part number

Fig 2. Flash sectors

Both the secondary USB bootloader and the user application reside in flash. Therefore, for the secondary USB bootloader to flash the user application without modifying any of its own code, the user application will be flashed starting with the next available sector.

Additionally, any flash sectors that do not contain any code may be used for data storage using the IAP commands.

3.2 ISP

ISP (In-system Programming) is supported in all LPC1700 series microcontrollers. ISP programming can be used to flash the microcontroller while it is in the end-user system. The ISP protocol consists of commands that are sent in ASCII format via the UART0 interface.

A detailed description of the ISP commands can be found in the LPC1700 user manual.

A free 3rd-party utility is available from Flash Magic at:
http://www.nxp.com/redirect/flashmagictool.com/

3.3 IAP

IAP (In Application Programming) is a feature that allows a user application to erase and write to on-chip flash memory. In order for the secondary bootloader to flash the user application onto the on-chip flash it needs to utilize these IAP commands.

A detailed description of the IAP commands can be found in the LPC1700 user manual.

© NXP B.V. 2009. All rights reserved.

3.4 CRP

The LPC1700 has three levels of CRP (Code Read Protection) as shown in Table 1.

Table 1. Code Read Protection (CRP)

CRP Explanation	CRP Level	Volume Label
The user flash can be read or written.	No CRP	CRP DISABLE
The user flash content cannot be read but can be updated. The flash memory sectors are updated depending on the new firmware image	CRP1	CRP1 ENABLE
The user flash content cannot be read but can be updated. The entire user flash memory is erased before writing the new firmware image.	CRP2	CRP2 ENABLE
The user flash content cannot be read or updated. The USB bootloader ignores the Entry Mechanism (Update Entry Pin) and always executes the user application if present. If user application is not present then "Update" mode is entered.	CRP3	CRP3 ENABLE

CRP is used as a security feature so that the code stored in flash can't be read using external tools, such as JTAG. The level of access restrictions depends on the CRP level selected.

Caution: If CRP3 is enabled, changes to the flash space can only be performed using IAP in the user application. Assuming a valid user application is present, the secondary USB bootloader will NOT enter ISP mode if CRP3 is selected.

4. USB communication

There are many USB Device Classes like DFU (Device Field Upgrade), HID (Human Interface Device), and MSCD (Mass Storage Class Device). The MSCD presents easy integration with a PC's operating system. This class allows the embedded system's flash memory space to be represented as a folder in a Windows/Linux environment, and the user can update the flash with a binary image using drag-and-drop (e.g., using Windows Explorer).

To make the LPC1700 appear like a folder (or disk drive), we need a FAT (File Allocation Table). In order to fully understand how the FAT file system works, the reader is advised to search the web for details on File Allocation Table and Storage Class Devices.

In our example code, we implemented the FAT12 system, which supports up to 32 MB (volume size of the drive), making this useful for large embedded devices. To simplify things, the LPC1700 on-chip code Flash will appear as one single entity (file name: firmware.bin), solving any defragmentation problems. FAT12 is supported by Windows 9x to XP, Vista and Linux.

In this application note, we do not attempt to explain how the Mass Storage Class is implemented. This secondary USB Bootloader code is a modification of Keil's USB Mass Storage Class example (http://www.nxp.com/redirect/keil.com/336.asp). The source files for the complete project are provided for the reader to use and understand.

© NXP B.V. 2009. All rights reserved.

5. Secondary USB bootloader

5.1 Entering ISP mode

The ISP mode of the secondary USB bootloader is the mode at which it flashes the user application onto the LPC1700.

In order to get into the ISP mode, the secondary USB bootloader follows the boot sequence shown below.

```
071 void main (void) {
072
073   /* If CRP3 is enabled and user flash start sector is not blank, execute the user code */
074   if( crp == CRP3 && user_code_present() )
075   {
076       execute_user_code();
077   }
078
079   /* check_isp_entry_pin() function does not return if isp entry is not requested */
080   if( user_code_present() )
081   {
082       check_isp_entry_pin();
083   }
084   /* User code not present or isp entry requested */
085   enter_usb_isp();
086 }
```

Fig 3. Secondary USB Bootloader boot sequence

5.1.1 Using an entry pin

The secondary USB bootloader will check the status of a GPIO pin to determine if the entry is valid. This is the easiest way since no post processing is needed. By default the bootloader uses P1.20.

5.1.2 Automatic ISP entry

If the secondary USB bootloader detects that no user application is present upon reset, then it will automatically enter ISP mode.

5.1.3 ISP entry disabled

If the secondary USB bootloader detects that a user application has already been installed and that CRP is set to level 3, then it will not enter ISP mode.

5.2 Bootloader size

Since the bootloader resides within user programmable flash, it is typically desired to have the bootloader to be designed as small as possible. The larger the secondary USB bootloader is the less flash space is available to the user application. Note that flash applies to a sector-by-sector basis.

By default, the USB bootloader has been designed to fit within the first two flash sectors (Sector 0-1) so that the user application can start from sector 2.

5.3 Code placement in flash

The secondary bootloader is placed at the starting address 0x0 so that it will be executed by the LPC1700 after reset. The USB communications within the bootloader utilizes interrupts and therefore its vector table is shown in Fig 4.

© NXP B.V. 2009. All rights reserved.

Since programming flash is based on a sector-by-sector basis, the code for the user application cannot be stored in any of the same flash sectors the bootloader uses.

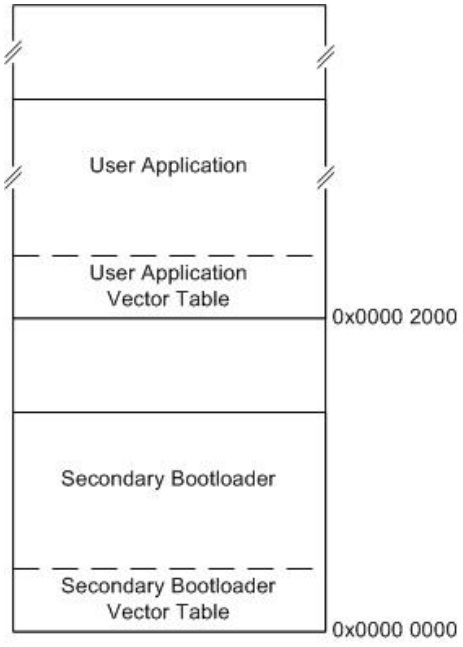


Fig 4. Bootloader and user application in flash

For efficient use of flash space, the user application should be flashed into the next available empty sector after the bootloader. Note that the user application will also contain its own interrupt vector table as shown in Fig 4.

5.3.1 User application execution

If the secondary USB bootloader is not going to enter ISP mode, it will start the execution of the user application. To do so, it will issue a branch statement to the application's starting address, which happens to be the starting address of that particular flash sector.

By default, the secondary USB bootloader will start user code execution by issuing a branch statement to 0x0000 2000.

5.3.2 Updating the vector table offset register

The secondary USB bootloader will change the Cortex-M3 NVIC's Vector Offset Register in case the user application uses interrupts.

```
/* Change the Vector Table to the USER_FLASH_START
in case the user application uses interrupts */
NVIC_SetVectorTable(NVIC_VectTab_FLASH, USER_FLASH_START);
```

Fig 5. Changing the NVICs VTOR register

© NXP B.V. 2009. All rights reserved.

Changing the vector table offset register will not affect the user application if it doesn't use interrupts. If the user application wishes to use its vector table in SRAM, it is up to the user application to create that vector table.

See ARM's technical reference manual on the Cortex M3 for more details.

5.4 Code read protection

The secondary USB bootloader supports all levels of CRP.

When the bootloader is in ISP mode, the MSC drive that it has created will show the CRP level. The drive (or volume) name can be seen from the "My Computer" window.

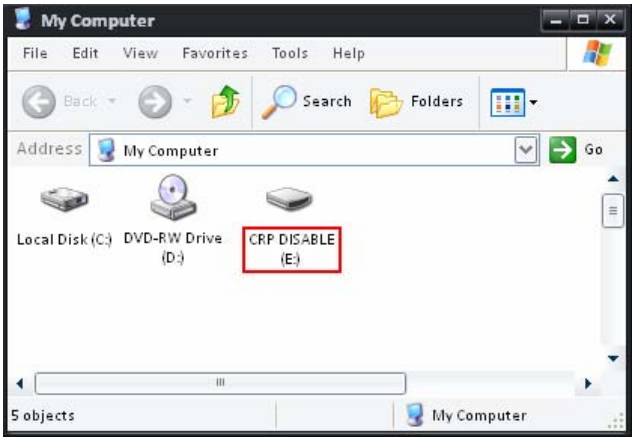


Fig 6. USB Bootloader MSCD – CRP Disabled

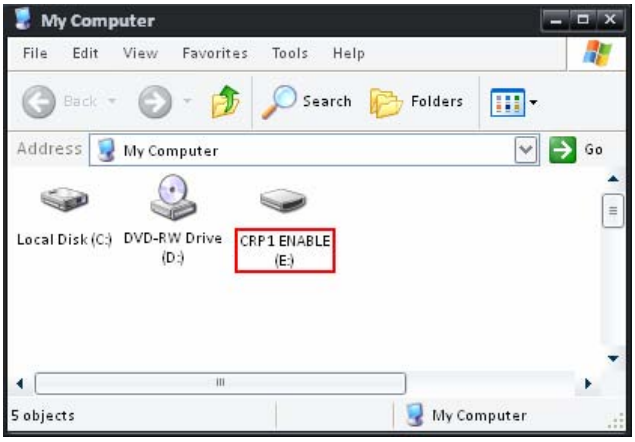


Fig 7. USB Bootloader MSCD – CRP1 Enabled

Opening the `firmware.bin` located on the MSC drive while CRP is enabled will show a file containing only NULL values. If CRP is disabled, then the binary file will contain the contents of the flash sector starting from the user application.

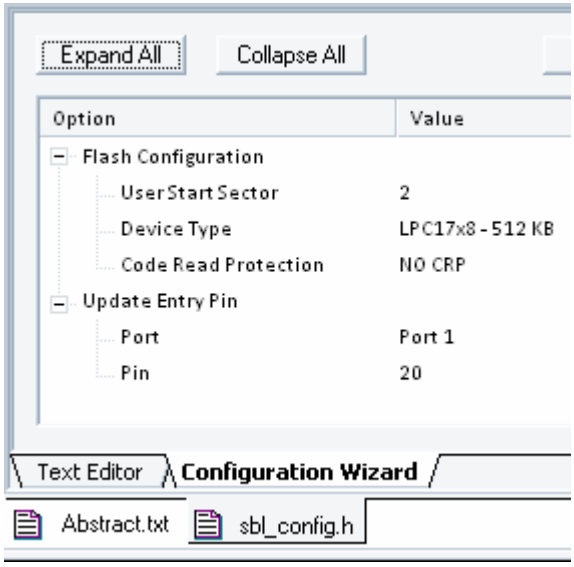
5.5 Bootloader code modifications

The secondary bootloader has several modification options available.

5.5.1 Flash configuration

The bootloader's flash configurations can be found in the `sbl_config.h` file. When using Keil's "Configuration Wizard" these configuration options can be easily changed.

© NXP B.V. 2009. All rights reserved.



(1) "Configuration Wizard" view of "sbl_config.h"

Fig 8. Secondary bootloader modification options

Table 2. Secondary bootloader flash configuration options

Flash Configuration Options	Range of values	Default value
User Start Sector	0 - 29	2
Device Type	8, 64, 128, 265, 512 KB	512 KB
Code Read Protection	NO CRP, CRP1, CRP2, CRP3	NO CRP

Table 3. Secondary bootloader update entry pin options

Update Entry Pin Options	Range of values	Default value
Port	Port 0, Port 1, Port 2, Port 3, Port 4	Port 1
Pin	0 - 31	20

5.5.1.1 User start sector

This field specifies at which sector number the user application should start from. Unless the bootloader is modified such that it requires more code space than the first two sectors (Sector 0-1) this field does not need to be modified. The starting address for sector 2 on the LPC1700 is 0x0000 2000.

5.5.1.2 Device type

The device type refers to a particular part LPC1700 part number. This field specifies how much flash is available on that device. This should be changed to reflect the intended part number for which the bootloader is built for.

© NXP B.V. 2009. All rights reserved.

5.5.1.3 Code read protection

Changing the code read protection field will enable/disable the LPC1700 CRP feature.

5.5.1.4 Entry pin

The “Port” and “Pin” fields specify which I/O pin is to be checked for ISP entry. The defaults are “Port 1” and “20” respectively.

5.5.2 USB configuration

The USB configurations can be found in the `usbcfg.h` file. When using Keil’s “Configuration Wizard” these configuration options can be easily changed.

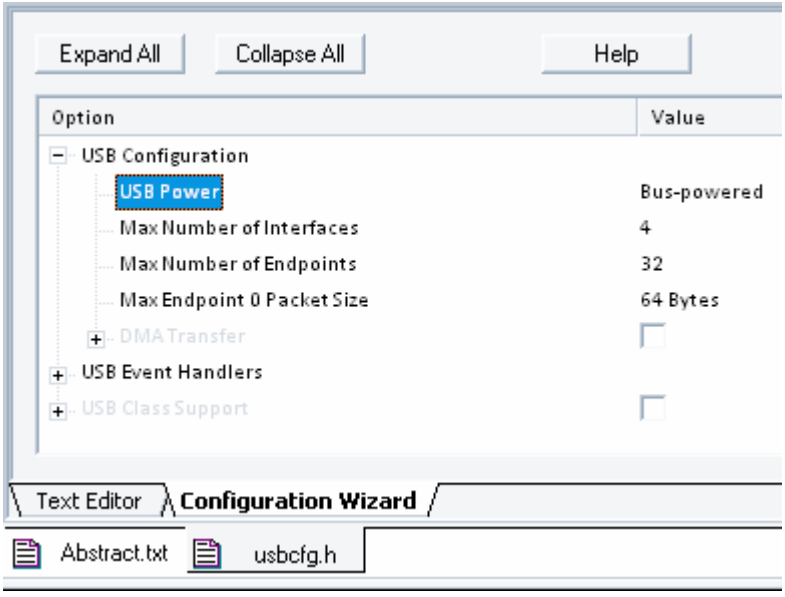


Fig 9. USB configuration

The default values shown are sufficient for the secondary USB bootloader example. Specific USB configuration properties are outside of the scope of this Application Note.

One option of interest however might be “USB Power”. For “USB Power” there are two possibilities: bus-powered and self-powered. The MCB1700 is bus-powered since it derives its power from the bus. If the user’s application has its own power source, then the self-powered option needs to be selected.

5.6 Using the USB bootloader

5.6.1 Installing the secondary USB bootloader

- Open the secondary USB bootloader sample project using Keil’s uVision.
- If necessary, make the desired code changes.
- Build the project.
- Erase the LPC1700.
- Flash the LPC1700 with the bootloader.
- Reset the LPC1700.

© NXP B.V. 2009. All rights reserved.

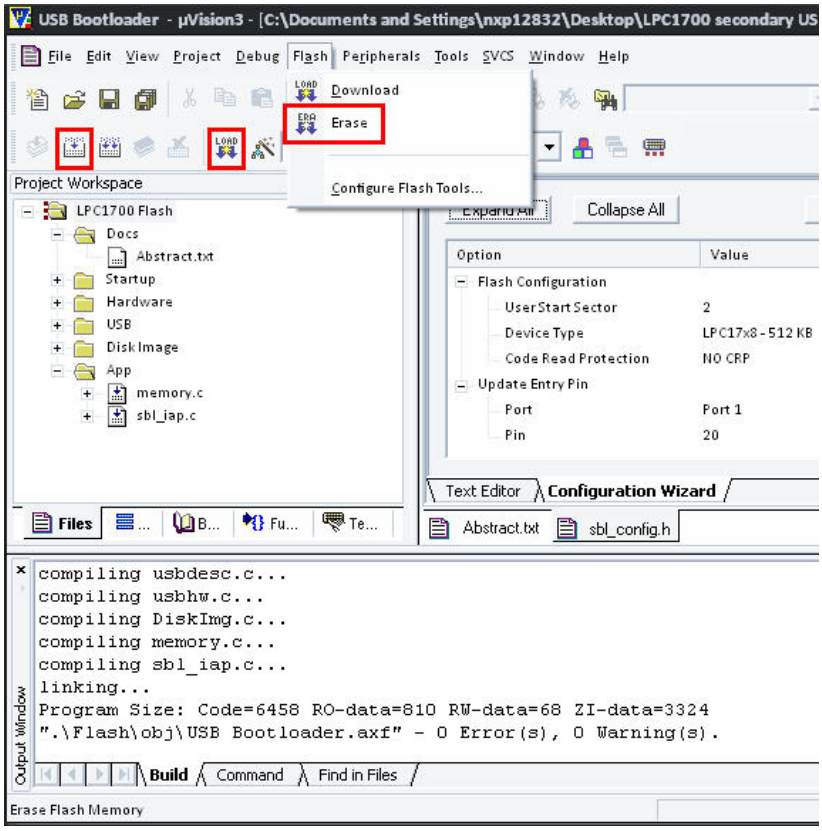


Fig 10. Flashing the secondary USB bootloader using Keil's uVision

At this point, because the LPC1700 has no user application it automatically enters ISP mode upon reset.

Go to the "My Computer" window and locate the newly created drive with a "CRP DISABLE" label. If the CRP level has been changed in USB bootloader project, it will display the new CRP level instead.

5.6.2 Flashing the user application

- Create the binary file of the user application (See Section 6).
- Enter the secondary USB bootloader's ISP mode.
- Open the newly created drive created in the "My Computer" window.

© NXP B.V. 2009. All rights reserved.

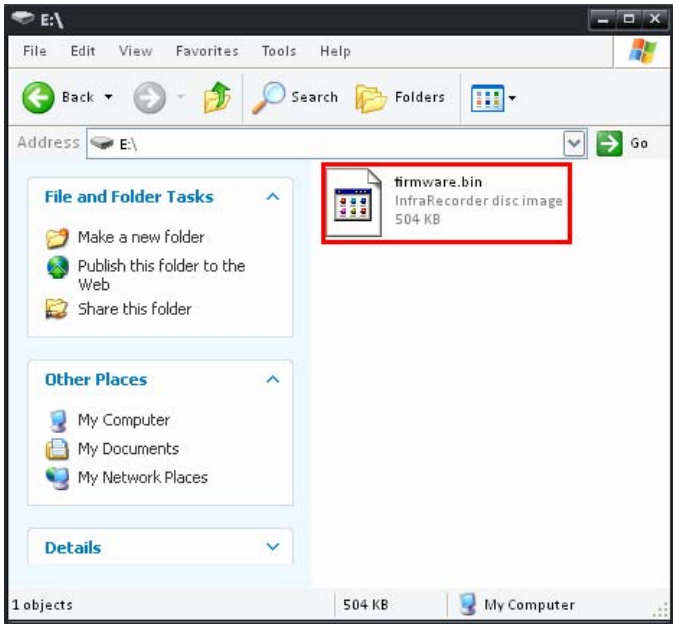


Fig 11. Firmware.bin in the USB bootloader's MSCD

- Delete `firmware.bin`
- Drag'n'Drop (or Copy and Paste) the binary file of the user application onto this drive.
- Reset the LPC1700

From here on, unless the entry pin (P1.20 by default) is grounded, the LPC1700 will boot into the user application.

6. User application

This section briefly describes the necessary steps needed in order to make the user application usable by the bootloader.

6.1 User code entry

To execute the user application the secondary USB bootloader will jump to a predetermined flash sector address. Therefore, the user application must be built so that it can run from that starting address. By default, this address is 0x0000 2000.

See Fig 4 for an illustration on how the user application is stored relative to the secondary USB bootloader.

6.1.1 Updating the starting address

- Open the user application's uVision project.
- Open the "Target Options"
- Change the starting address to "0x2000"

© NXP B.V. 2009. All rights reserved.

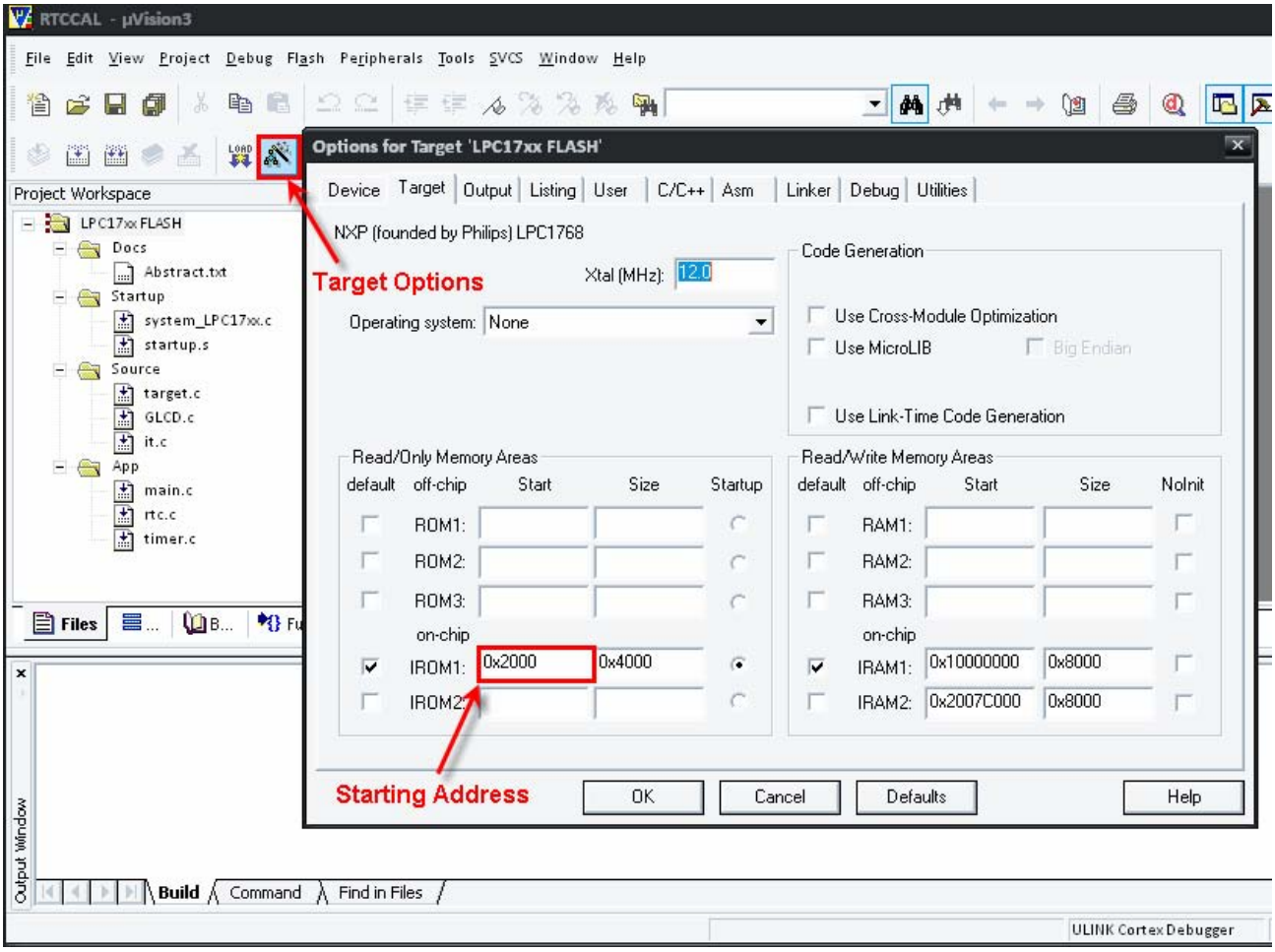


Fig 12. Configuring user application's starting address

6.1.2 Creating the binary file

By default, Keil's uVision does not create the binary required by the secondary USB bootloader. In order to create this binary, we need to use an external command line driven tool that comes included with Keil's toolchain. Depending on your project configuration, you may have to change the path for the tool's command line arguments.

- Select the "User" tab in the "Target Options"
- Check the "Run #1:" from the "Run User Programs After/Rebuild"
- Type in the command to run the `fromelf` tool.

© NXP B.V. 2009. All rights reserved.

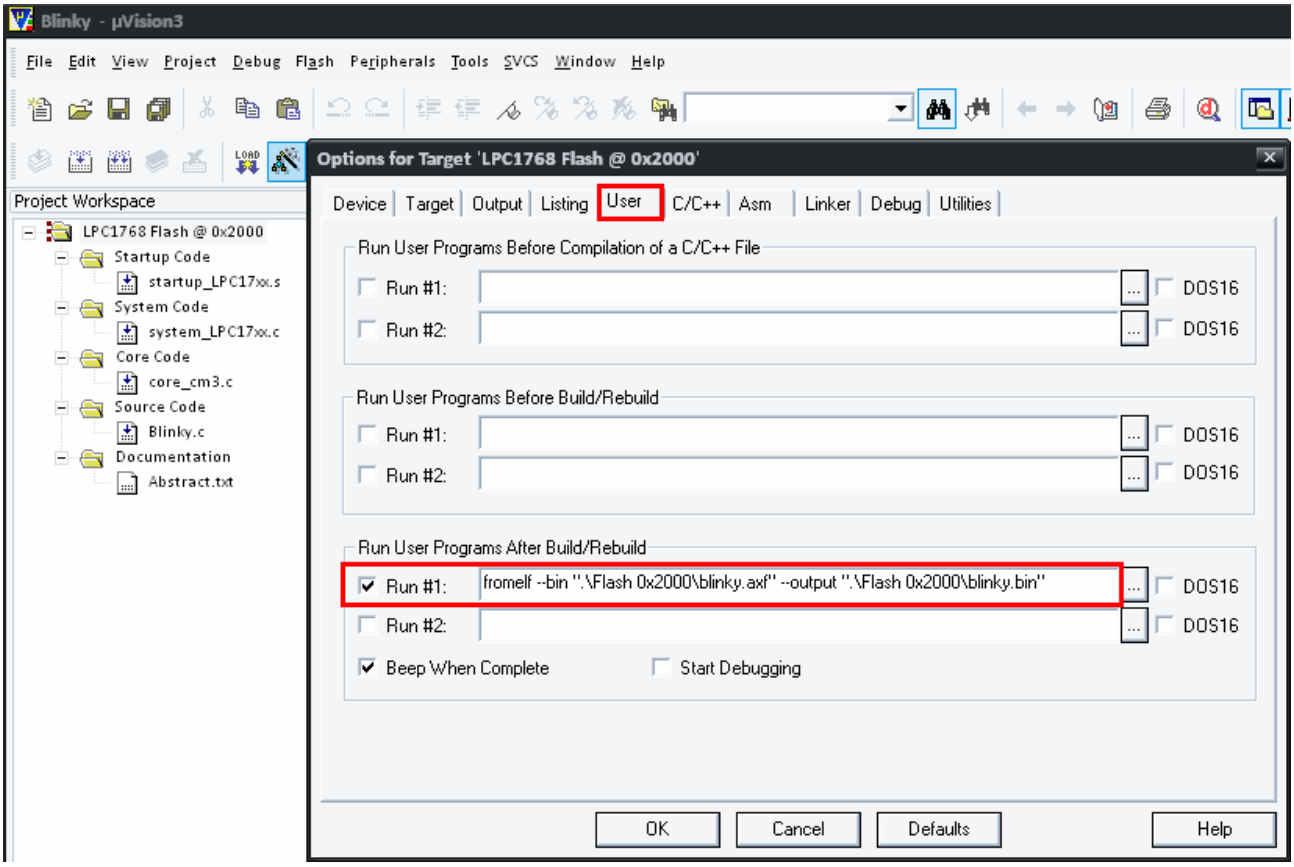


Fig 13. Configuring "fromelf" to create the required binary file

6.2 Interrupt vectors

The secondary USB bootloader will automatically adjust the NVIC's vector table offset register (VTOR) to point to same starting address at which it branches to for user code execution. This is done so that there isn't any extra overhead for the user when he or she is building the user application.

6.3 Startup file modification

The secondary USB bootloader already handles the CRP configuration (See Section 5.5.1). Therefore, if your user application uses the `system_LPC17xx.s` startup file from one of Keil's application examples, some modifications may be recommended.

```
107
108 ;                   IF      :LNOT::DEF:NO_CRP
109 ;                   AREA    |.ARM.__at_0x02FC|, CODE, READONLY
110 ;CRP_Key            DCD     0xFFFFFFFF
111 ;                   ENDIF
112
```

Fig 14. Commented CRP configuration block of the "startup_LPC17xx.s" file

© NXP B.V. 2009. All rights reserved.

By design, this block of code (Fig 14) from the system_LPC17xx.s sets the user application's CRP level because it is intended to be used without a secondary bootloader.

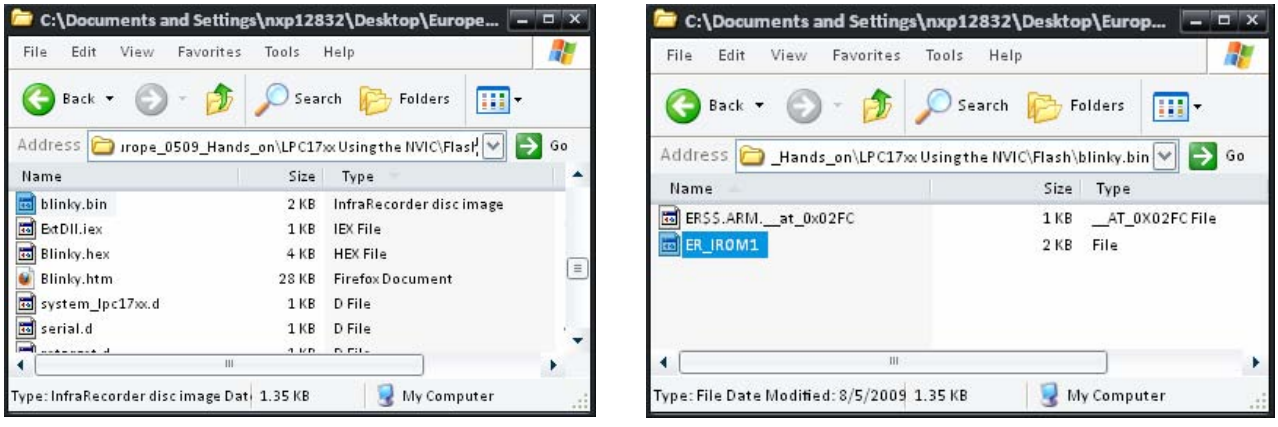
```
; *************************************************************
; *** Scatter-Loading Description File generated by uVision ***
; *************************************************************

LR_IROM1 0x00002000 0x0007E000  {    ; load region size_region
  ER_IROM1 0x00002000 0x0007E000  {  ; load address = execution address
   *.o (RESET, +First)
   *(InRoot$$Sections)
   .ANY (+RO)
  }
  RW_IRAM1 0x10000000 0x00008000  {  ; RW data
   .ANY (+RW +ZI)
  }
}
```

(1) This scatter file reflects the Target option changes to relocate code to sector 2 (0x2000)

Fig 15. uVision's linker scatter file

As the starting address for the user application was adjusted in section 6.1.1, uVision's scatter file has also been updated to reflect these adjustments. However, since the CRP code block inside the system_LPC17xx.s startup file explicitly addresses 0x02FC, the linker detects that this portion of code is outside of the scatter file's load region and hence automatically creates an additional load region.

Without removing the CRP code block, the fromelf tool will output a folder containing multiple binaries instead of a single binary file due to the multiple load regions.



a. Without the CRP code block

b. With the CRP code block

Fig 16. User application binary outputs

© NXP B.V. 2009. All rights reserved.

The binary file referencing to “0x02FC” in Fig 16b contains the code block from the system_LPC17xx.s startup file. By commenting out the code in Fig 14 you should only see only one binary file as shown in Fig 16a.

When comparing the HEX (.hex) files of the user application without the CRP code with the user application with the CRP code, the difference can be clearly observed. Fig 17 shows the two hex files side-by-side. The machine code on left has been generated without CRP code.

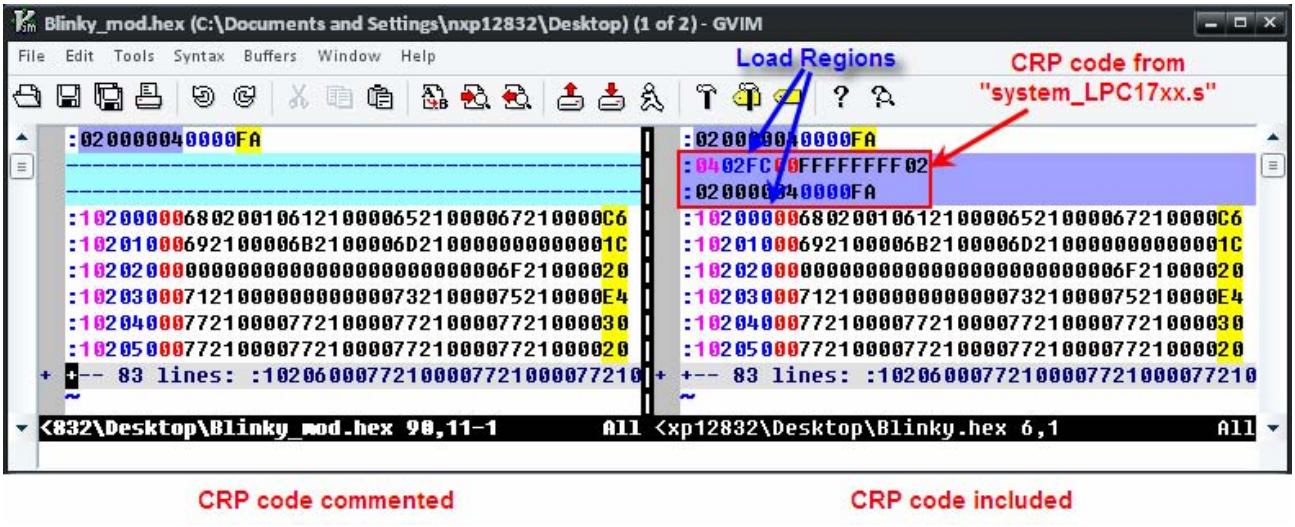


Fig 17. View the HEX files of the user application

The machine code on the right is the exact same code, with the exception that it includes the CRP code located in the system_LPC17xx.s startup file. In this build, you can see that it includes data intended to be placed at 0x02FC.

7. Conclusion

By using a secondary bootloader it is possible to conveniently perform in-application software updates without using additional development hardware, such as JTAG. In this case this secondary bootloader uses USB to transfer the binaries to the LPC1700, but other channels such as Ethernet and I2C are also possible.

This application note serves as a reference on how use, create, and modify a secondary USB bootloader. The secondary USB bootloader has been designed as a standalone project that contains all of its dependant source files. The user applications used for this demonstration are modified code examples that come included with Keil’s uVision toolchain.

© NXP B.V. 2009. All rights reserved.

8. Legal information

8.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

8.2 Disclaimers

General — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in medical, military, aircraft, space or life support equipment, nor in applications where failure or malfunction of a NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is for the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from national authorities.

8.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

© NXP B.V. 2009. All rights reserved.

9. Contents

1. Introduction 3
2. Requirements 3
2.1 Hardware requirements 3
2.1.1 Programming 4
2.1.2 Jumper settings 4
2.1.3 USB cable 4
2.1.4 Bootloader's ISP entry mechanism 4
2.2 Software requirements 4
3. Flashing the LPC1700 4
3.1 Flash sectors 4
3.2 ISP 5
3.3 IAP 5
3.4 CRP 6
4. USB communication 6
5. Secondary USB bootloader 7
5.1 Entering ISP mode 7
5.1.1 Using an entry pin 7
5.1.2 Automatic ISP entry 7
5.1.3 ISP entry disabled 7
5.2 Bootloader size 7
5.3 Code placement in flash 7
5.3.1 User application execution 8
5.3.2 Updating the vector table offset register 8
5.4 Code read protection 9
5.5 Bootloader code modifications 9
5.5.1 Flash configuration 9
5.5.1.1 User start sector 10
5.5.1.2 Device type 10
5.5.1.3 Code read protection 11
5.5.1.4 Entry pin 11
5.5.2 USB configuration 11
5.6 Using the USB bootloader 11
5.6.1 Installing the secondary USB bootloader 11
5.6.2 Flashing the user application 12
6. User application 13
6.1 User code entry 13
6.1.1 Updating the starting address 13
6.1.2 Creating the binary file 14
6.2 Interrupt vectors 15
6.3 Startup file modification 15
7. Conclusion 17
8. Legal information 18
8.1 Definitions 18
8.2 Disclaimers 18
8.3 Trademarks 18
9. Contents 19

Please be aware that important notices concerning this document and the product(s) described herein, have been included in the section 'Legal information'.

© NXP B.V. 2009. All rights reserved.

For more information, please visit: http://www.nxp.com
For sales office addresses, email to: salesaddresses@nxp.com

Date of release: 8 September 2009
Document identifier: AN10866_1

founded by
PHILIPS